|  | **2, LEBUH ACHEH, GEORGE TOWN**  **10300 GEORGE TOWN**  **PULAU PINANG**  **INFORMATION SHEET** | |
| --- | --- | --- |
| **PROGRAM’S CODE & NAME** | J620-002-4:2020 FRONT-END SOFTWARE DEVELOPMENT | |
| **LEVEL** | FOUR (4) | |
| **COMPETENCY UNIT NO. AND TITLE** | J620-002-4:2020-C04 MOBILE APPLICATION WITH THIRD PARTY API DEVELOPMENT | |
| **WORK ACTIVITIES NO. AND STATEMENT** | 1. **CREATE MOBILE APP DESIGN MOCK-UP ELEMENTS.** 2. PLAN MOBILE APP DESIGN STRUCTURE. 3. TRANSFORM MOCK-UP TO MOBILE APP. 4. INTEGRATE MOBILE APP WITH DATA SOURCE. 5. VERIFY SUCCESSFUL API INTEGRATION 6. VERIFY DEVELOPED MOBILE APP. 7. VERIFY MOBILE APP ACCESSIBLE GLOBALLY. | |
| **CODE NO.** | J620-002-4:2020-C04/IS(1/15) | Page: 1 of |

**TITLE**:

**INTRODUCTION OF MOBILE APP PROGRAMMING**

**PURPOSE**:

This information sheet is intended to provide insight and knowledge to trainees with regards to the fundamentals of mobile app programming and user interface.

**INFORMATION:**

This information sheet provides useful notes and explanations on fundamental concepts and foundations for understanding mobile applications, how to develop it and designing a good user interface.

# **INTRODUCTION TO MOBILE APP PROGRAMMING**

Mobile app or mobile application is a type of software application that is intended to run specifically on mobile devices. These applications serve to provide similar functionality to those available on full-fledged computers. Mobile apps are generally small and individually have a limited set of capabilities.

Mobile applications can be categorised by technologies used into 3 types: native, web and hybrid. Table 1 shows the differences between the types.

Table 1: Types of mobile application

| Type | Description |
| --- | --- |
| Native apps | Created for one specific platform or operating system |
| Web apps | Responsive versions of websites that can work on any mobile device or OS because they are delivered using a mobile browser. |
| Hybrid apps | Combinations of both native and web apps, but wrapped within a native app, giving it the ability to have its own icon or be downloaded from an app store. |

## Native applications

Native apps are built specifically for a mobile device’s operating system. Thus, you can have native Android mobile apps or native iOS apps, not to mention all the other platforms and devices. Because they are built for just one platform, you cannot install and use apps developed for one platform on another platform. For example, you cannot install an Android app in iOS devices and vice versa.

Because of their singular focus, native apps have the advantage of being faster and more reliable in terms of performance. They are generally more efficient with the device’s resources than other types of mobile apps. Native apps utilize the native device UI, giving users a more optimized customer experience. And because native apps connect with the device’s hardware directly, they have access to a broad choice of device features like Bluetooth, phonebook contacts, camera roll, NFC, and more.

However, the problem with native apps lies in the fact that if you start developing them, you have to duplicate efforts for each of the different platforms. The code you create for one platform cannot be reused on another. This drives up costs. Not to mention the effort needed to maintain and update the codebase for each version. And then, every time there is an update to the app, the user has to download the new file and reinstall it. This also means that native apps do take up precious space in the device’s storage.

## Web applications

Web apps behave similarly to native apps but are accessed via a web browser on your mobile device. They are not standalone apps in the sense of having to download and install code into your device. They are responsive websites that adapt its user interface to the device the user is on. In fact, when you come across the option to “install” a web app, it often simply bookmarks the website URL on your device.

One kind of web app is the progressive web app (PWA), which is basically a native app running inside a browser.

Because it is web-based, there is no need to customize a platform or OS. This cuts down on development costs. Plus, there is nothing to download. They will not take up space on your device memory like a native app, making maintenance easier – just push the update live over the web. Users do not need to download the update at the app store.

On the other hand, web apps are entirely dependent on the browser used on the device. There will be functionalities available within one browser and not available on another, possibly giving users varying experiences. And because they are shells for websites, they will not completely work offline. Even if they have an offline mode, the device will still need an internet connection to back up the data on your device, offer up any new data, or refresh what is on screen.

## Hybrid applications

Hybrid apps are web apps that look and feel like native apps. They might have a home screen app icon, responsive design, fast performance, even be able to function offline, but they are really web apps made to look native. Hybrid apps use a mixture of web technologies and native APIs.

Building a hybrid app is much quicker and more economical than a native app. As such, a hybrid app can be the minimum viable product – a way to prove the viability of building a native app. They also load rapidly, are ideal for usage in countries with slower internet connections and give users a consistent user experience. Finally, because they use a single code base, there is much less code to maintain.

# **MOBILE OPERATING SYSTEM**

Mobile applications run on top of mobile operating systems installed on the devices. As of 2021, the biggest platform for mobile application is Android. There are more than a billion devices that operate with Android Operating Systems and the number is increasing year by year.

## Android

The Android operating system was founded by Rich Miner, Nick Sears, Chris White, and Andy Rubin in 2003. Initially, the operating system was developed to empower digital cameras. But, as the market for digital cameras started to decline at that time, the company decided to bring the operating system into mobile phones. Believing in Android’s huge potential for the future of mobile computing, Google came in acquiring the founding company in 2005. In 2018, the first Android smartphone was announced to the public; the HTC Dream also known as T-Mobile G1 in the United States was powered by Android 1.0. Ever since that, Android has evolved into a gigantic platform that hovers around 84% of today's smartphone market share.

Android which is currently owned by Google has its own library of first-party and third-party apps called Google Play store. Through it, users can search, install, and use various types of apps that take advantage of their smartphone capabilities. In 2014 alone, there were over 1,100,000 applications in the Google Play store and the number tripled by the end of the first quarter in 2018.

## iOS

iOS is Apple's mobile operating system that runs on iPhone, iPad, and iPod Touch devices. Originally known as the iPhone OS, the name was changed with the introduction of the iPad. It uses a multi-touch interface in which simple gestures operate the device, such as swiping your finger across the screen to move to the next page or pinching your fingers to zoom out. There are more than 2 million iOS apps available for download in the Apple App Store, the most popular app store of any mobile device.

iOS popularised functions like zooming in and out as well as the feature that has been around so long now that many cannot believe it did not always exist – the swiping gesture. Over the course of 13 years, iOS has greatly developed in several areas. The App Store has become more prolific than many would have imagined, iOS took swiping even further by ditching the physical home button and deeper integration with Apple’s own apps like Apple Music, Podcasts, TV and more.

# **MOBILE APP PROGRAMMING**

There are multiple numbers of ways to develop a mobile application. From cross-platform framework to platform specific tool to language specific technologies. For example, to develop Android applications, programming languages such as Java and Kotlin is used. This is in contrast to iOS application development that primarily relies on programming languages such Swift and C-Objective.

## Android applications

Commonly for Android application development, an Integrated Development Environment (IDE) called Android Studio will be used. Android Studio is essentially an interface where you can enter your code (primarily Java or Kotlin) and access all the different tools necessary for development and it was developed by Google. Android Studio allows you to access libraries and APIs from the Android SDK, thereby giving you access to native functions of the operating system. You will also be able to build your app into an APK using Gradle, test it via a “virtual device” (emulator), and debug your code while it runs.

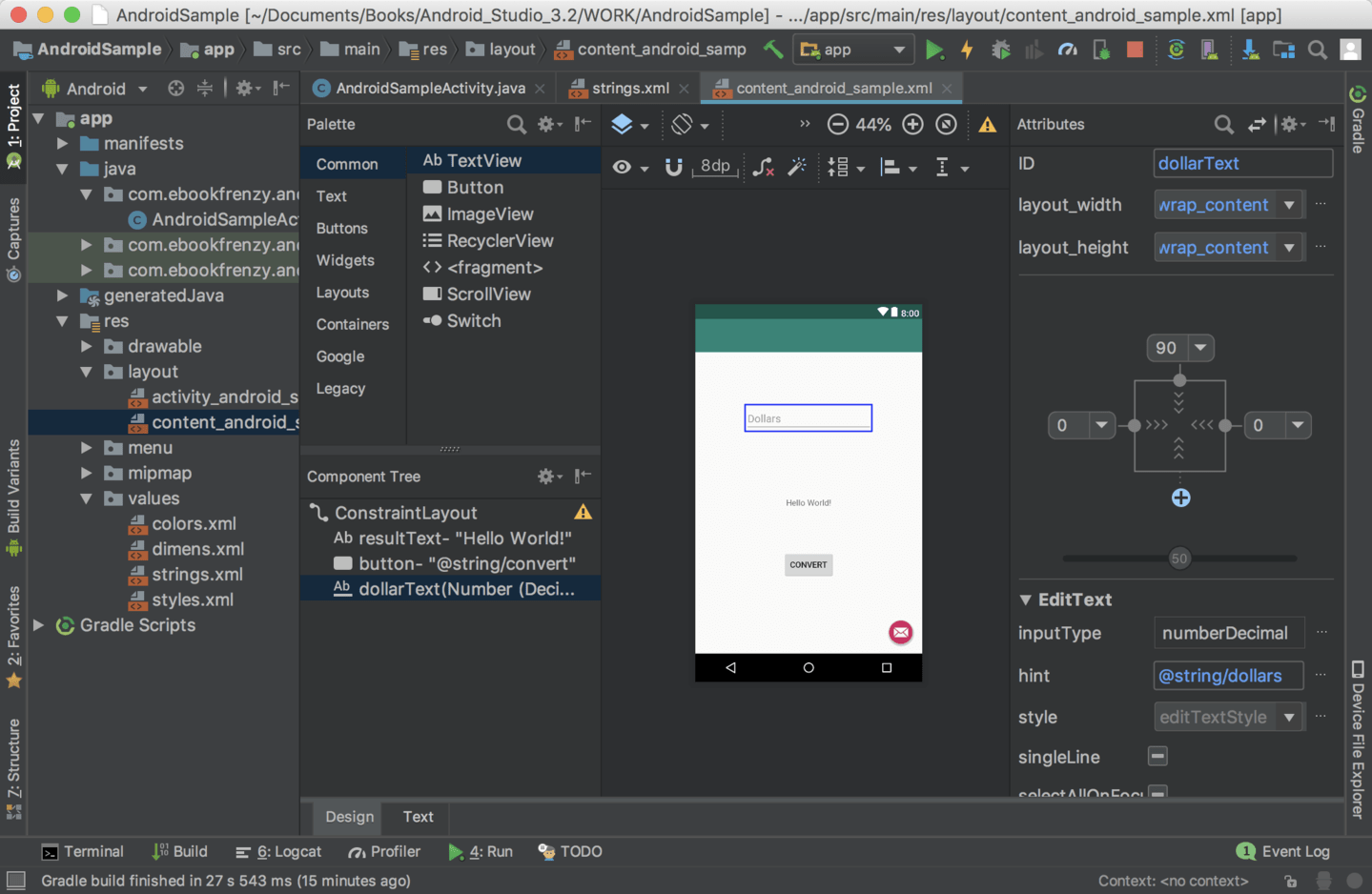


Figure 1: Android Studio interface

In order to develop apps using Android Studio, you can either use Windows, Mac, or Linux computer. Ideally, a test device running Android will be significantly helpful to test app performance on real world devices. In order to publish your app to Google Play Store, you will need a developer account which cost 25 USD for a lifetime.

## iOS applications

iOS runs specifically on Apple devices such as iPhone, iPad, and iPod. Primarily, iOS app developers will use Xcode, an Integrated Development Environment (IDE) developed by Apple. Similar to Android Studio, Xcode offers access to all the different tools, libraries, and APIs from the iOS SDK. Programming languages such as Swift and Objective-C is the major language used in iOS app development process.

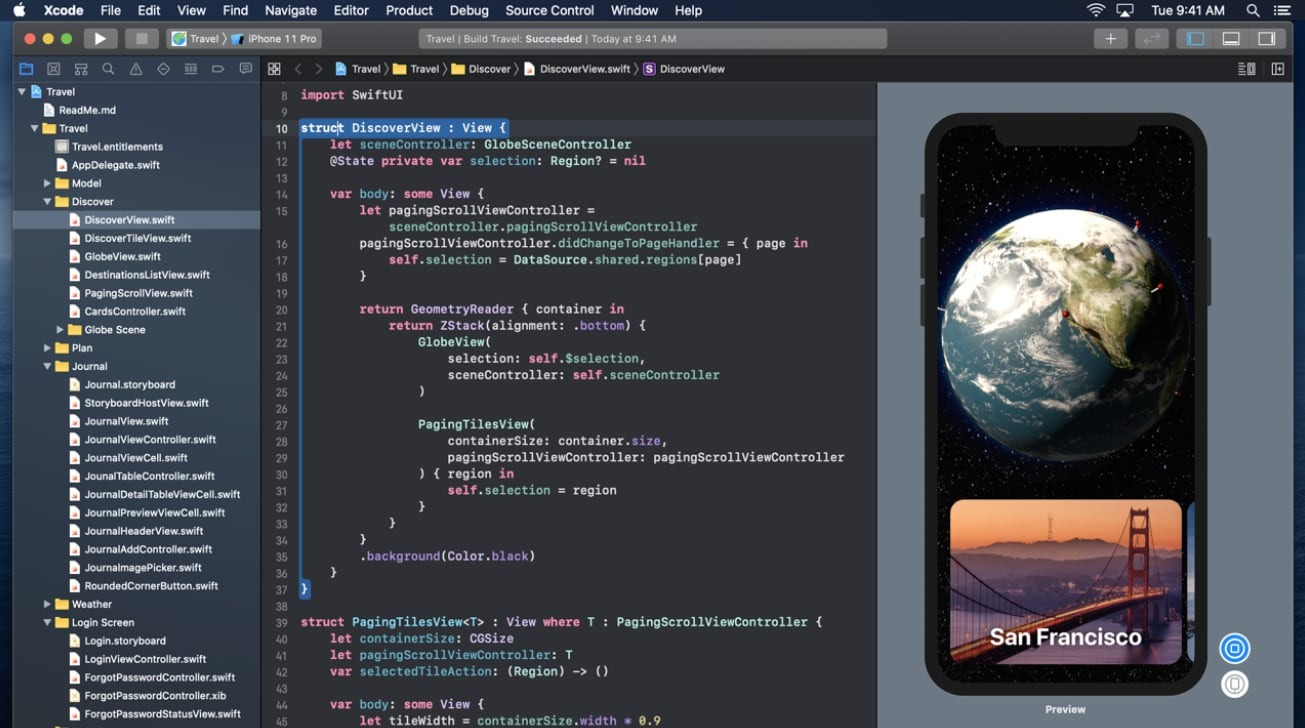


Figure 2: Xcode interface

In contrast, Xcode is only available on Mac computers. Which means there is no way around to start iOS app development besides getting Mac computers manufactured by Apple. You also need to register as an app developer license cost 99 USD annually to publish apps in the iOS app store.

## Hybrid applications

Since the hybrid app is basically a website that acts like a native app either Android or iOS in a native web container, there are multiple ways to approach it. First, you could start by developing the website to have the look and feel of a mobile app independently and then port it into a native app developed using Android Studio or Xcode. Another way, you could use a cross platform framework that will compile your code into both Android and iOS automatically. For example, Ionic framework offers a library of mobile-optimized UI components, gestures, and tools that allow developers to ship apps to the app stores with a single code base.

Ionic uses front-end technologies like HTML, CSS, JavaScript, and Angular for application development. Using web technologies, Ionic helps to build cross-platform mobile applications with a single codebase. Basically, it allows web developers to create web pages that are run inside a device’s browser instance called WebView. WebView may come as a plugin, and it is essentially an application component that renders web pages and displays them as a native application.

The first versions of Ionic were based on Angular, which is a popular front-end framework used for building dynamic web pages and progressive web applications, PWA for short. Ionic can use Angular CLI (Command-Line Interface) and components to create fully functional mobile applications.

For Ionic to access native functionality, Apache Cordova plugins is used. Cordova is a tool for building mobile applications using web technologies, relying on its own APIs instead of platform-specific ones. As long as Ionic uses WebView, it does not have access to the device’s hardware APIs by default. Cordova provides those APIs packed as plugins to gain access to functions like a smartphone’s camera, gyroscope, or sensors. These sets of APIs are also known as Cordova Bridge. Apache Cordova provides Ionic apps with access to native APIs, serving as a bridge between the web container (webview) and the device’s operating system.

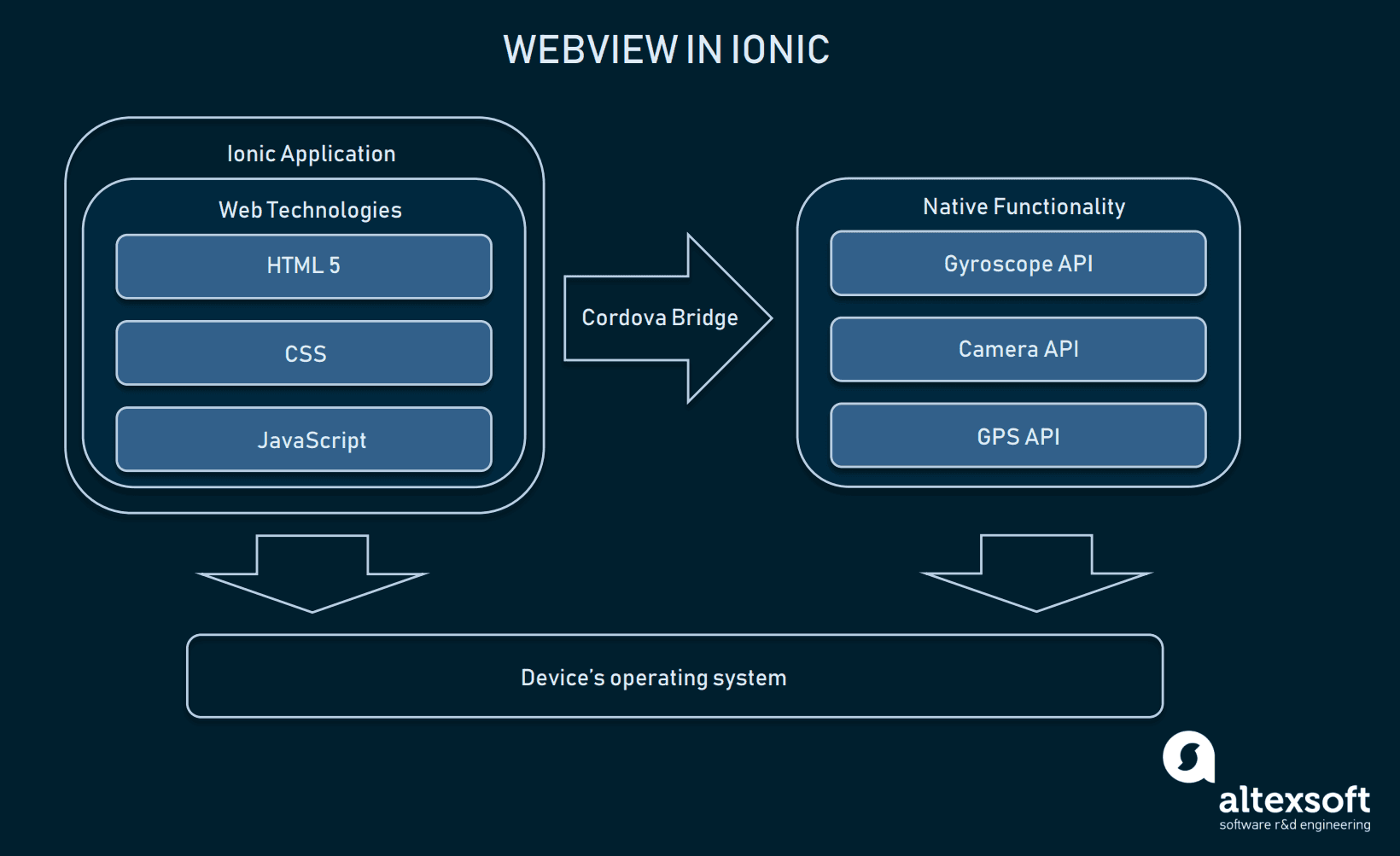


Figure 3: Hybrid and Ionic application working principles.

## React Native

React Native is a JavaScript framework for developing mobile applications that can run natively on both Android and iOS. It is based on ReactJS, developed at Facebook, which is a declarative, component-based framework for developing web user interfaces. Like ReactJS, React Native uses JSX, an XML markdown language for developing UIs which replaces HTML and CSS. From JSX, Ul components are compiled into native platform-specific components which create a fast and familiar experience for end-users.

Compared to hybrid apps developed with the Ionic framework, apps that developed using React Native is a native app. Rather than hosting a website in a web container, React Native compiles the source code written in JSX into specific source code on the platform, Android, and iOS. By using React Native, your web app can share the same logic as your mobile apps, maintaining consistency across platforms, and save you time and money. Instead of having to maintain multiple codebases and employ developers that are proficient in both Android and iOS, you can build a team of JavaScript developers that work on a single codebase.

As React Native uses underlying web technology, screens in the application can be reloaded reflecting code changes nearly instantly throughout development. This is compared to iOS or Android apps that typically have to recompile the codebase, re-launch their applications, and then navigate back to that screen to see their changes. JavaScript is the world’s most used programming language today. The benefits of having a vast array of open-source tools and libraries to reuse is hard to overstate.

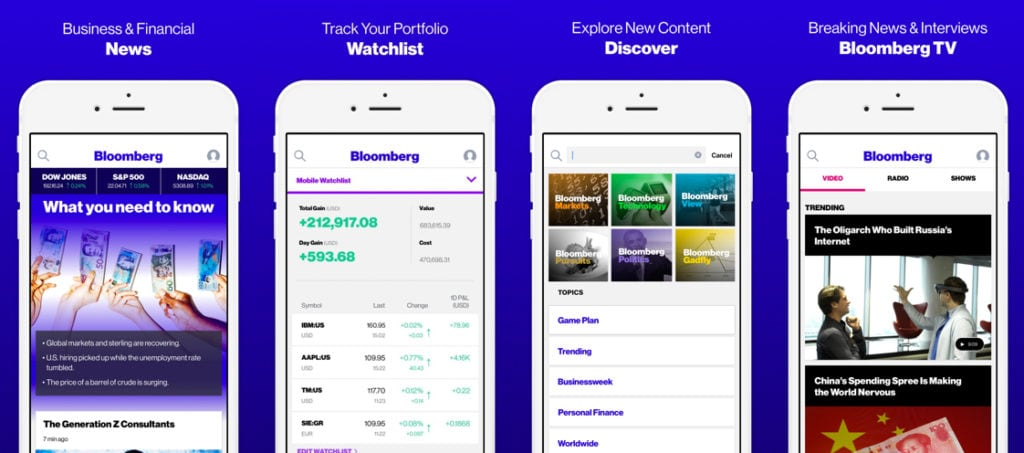


Figure 4: Bloomberg iOS and Android apps are built with React Native.

## Flutter

Maintained by Google, The Flutter SDK is a toolkit created to intelligently design applications for all types of screens and devices, allowing for cross-platform development. The toolkit competes with other native compatibility frameworks like React Native.

Flutter comes with a large collection of pre-built widgets (rows, columns, stacks, padding, centre, etc.), making it much easier to design and layout your mobile application. This means that instead of having to have separate codebases for your iOS app and Android app, you can utilize Flutter and have one codebase for all the different platforms that you want your application to support. Some companies that use Flutter are Google, Groupon, Alibaba, Square, eBay, and more.

Widgets are the basic building blocks of the Flutter UI. Unlike many other frameworks that separate views, layouts, view controllers, and other elements, Flutter offers a consistent, unified object model: the widget. Like components in React, widgets form a hierarchy and can be nested within each other. Furthermore, widgets inherit properties from their parents. Some widgets that come pre-built with the Flutter toolkit are text, button, row widget, column widget, and image widget. The list goes on. Each widget is customizable, so you can pass in certain properties to determine how it is displayed.

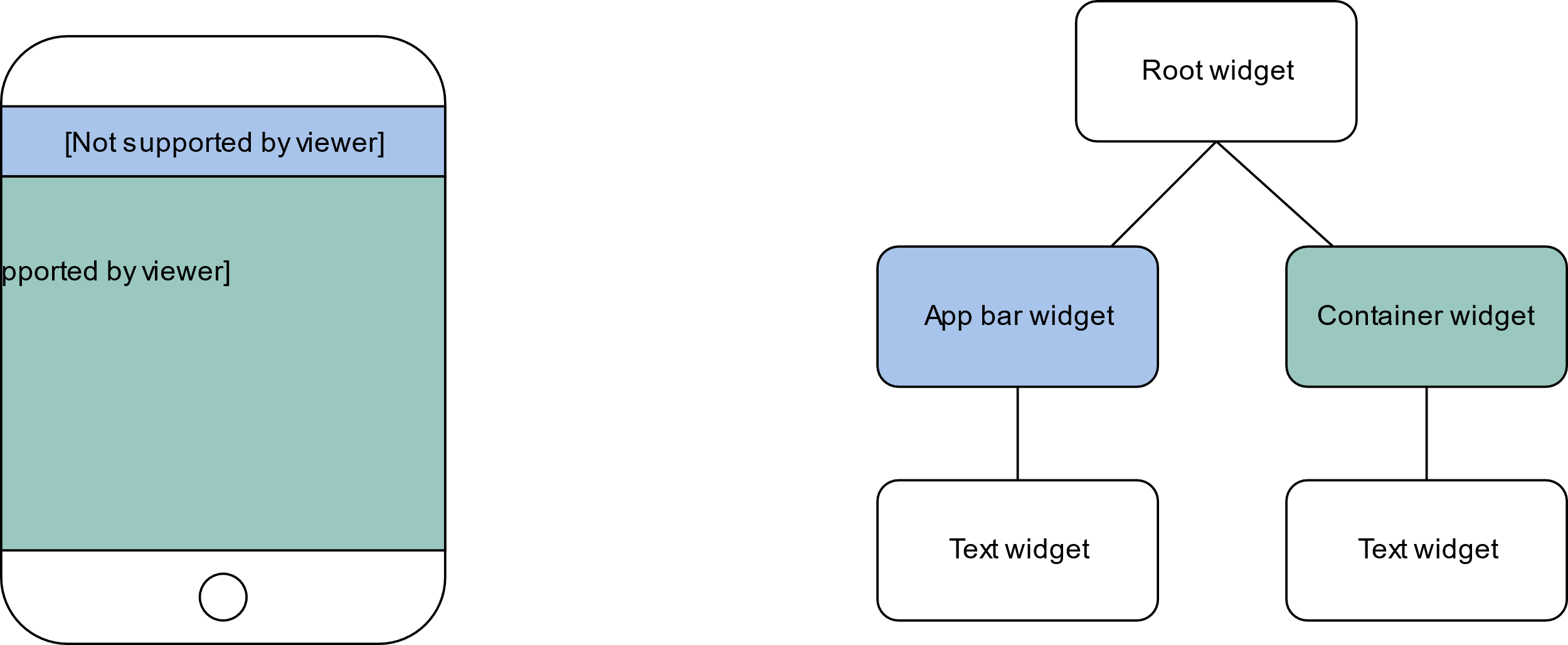


Figure 5: Flutter Widget Hierarchy

## Firebase

Firebase is one of the most trending technologies that help developers to make a web app or mobile apps in a short span of time. Firebase is a BASS (backend as a service) provided by Google, by means of the backend as a service means that you do not have to think about how to store the data in the backend and how to configure your service. All that stuff will be handled by the platform.

Firebase provides all the tools and the stuff that is needed to make web apps or mobile applications. Some of the services that usually need to develop or configure while making some apps are to configure servers, manage hosting, manage user authentication, write the crud operation logic for our app, storage, push notifications and the list goes on. Firebase has all these services, and you can get all this on the single click.

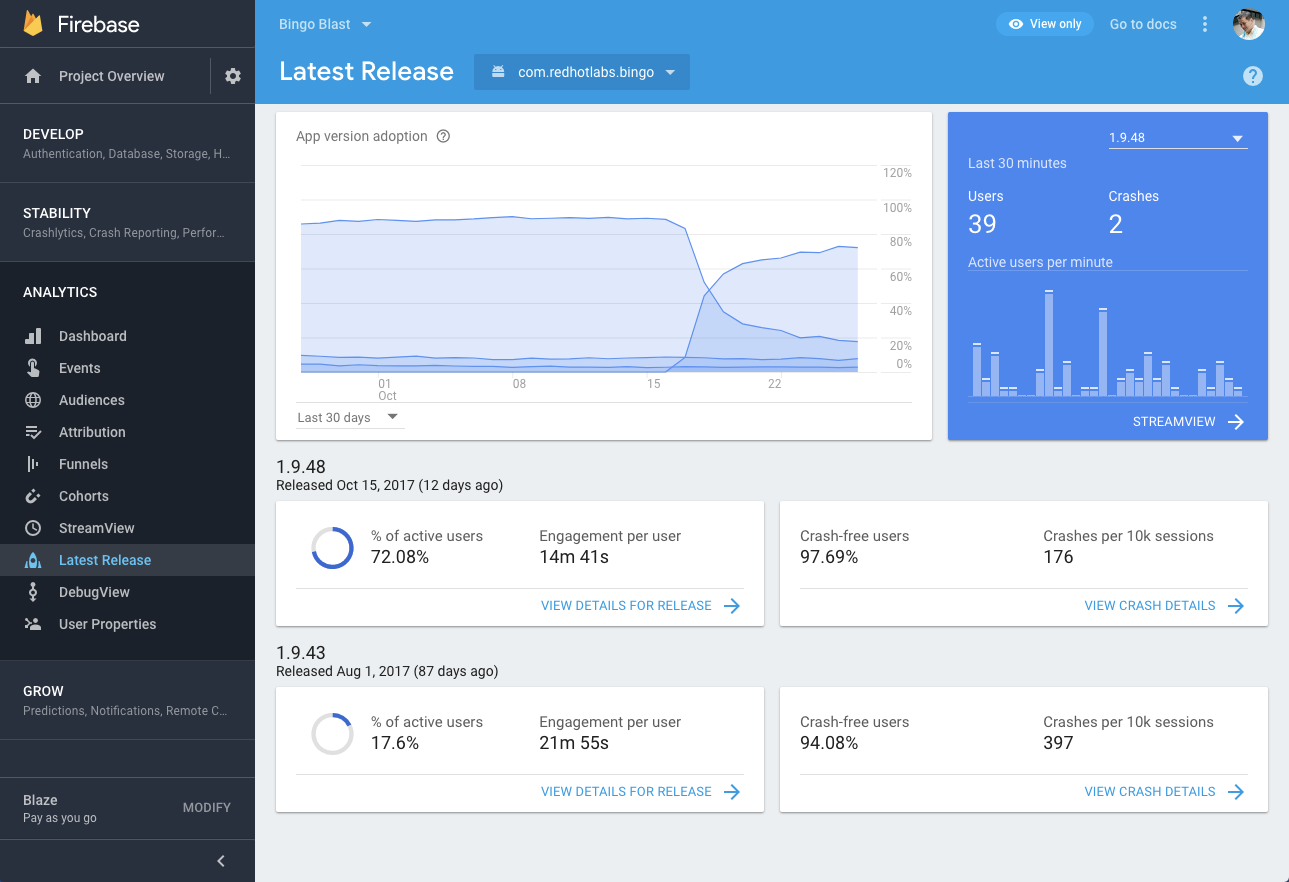


Figure 6: Firebase Analytics Interface

## Unity

Unity is widely known as a game engine and integrated development environment used by millions of developers to bring games to Android, iOS, Windows, consoles, and many more platforms. It is currently the most popular game engine on the Android platform, thanks to its streamlined interface and workflow, as well as its rich array of features and excellent versatility.

While Unity is primarily aimed at game development, it is not limited to games only. Although most of its features are designed with game development in mind, there are also a number of powerful features that might be useful for developing non-game apps in Unity. These are primarily graphical features, so if you want to include 3D elements in your app, Unity could be a very good choice.

# **USER INTERFACE FUNDAMENTAL**

App design combines the user interface (UI) and user experience (UX). While UI lends itself to the overall style of the app (including the colours, fonts, and general look and feel), UX focuses on the actual functionality and usability. A large number of users abandon an app after using it for the first time. Since users are picky about which apps they use and so quick to abandon those they do not enjoy, it is essential to invest time and effort in creating a great user experience. The better the design, the better the chance that a user will engage with it and thus keep using it.

Just like any other product or service, apps need to be designed, organized, and strategized to solve user problems. Before you begin the app design, it is important to conduct market research to gain a better understanding of your users. What do users need? What problem are you solving for them? What value will you bring them? Your app’s success depends on knowing these answers.

Great design is not only about great aesthetics; it is also about matching user expectations. You need to ensure the information architecture of your app matches the user’s mental model, navigation patterns are self-evident, all touch targets (such as buttons) are finger-friendly, and the app content looks equally good on mobile devices as it does on the web at various screen sizes and resolutions.

Though the answer to “what is good UI design” is rather tricky and subjective, there are UI design fundamentals that you can follow to design the app UI. Following are 7 UI design fundamentals:

## Defining the user

The goal is to provide users with value and fill a need. During the user interface design process, keep the user and their needs in mind at all times. Consider what interfaces they prefer, elements, typography, styles, and what calls to actions motivate them. Observing how they use those interfaces will give you invaluable insight.

Do not get caught up in trendy design styles, adding tons of features, or other unnecessary clutter that will distract users. Keep things conversational, by providing clear and concise labels for actions that sound like themselves and/or their peers. Focusing on your user first will enable you to create an interface that is appealing and leads them to the end goal. Familiar UI patterns

## Familiar UI patterns

There is no need to reinvent the wheel. What interfaces do your users spend the majority of their time on? Considering using popular interfaces, such as Facebook, Instagram, Google, and Gmail as UI design examples to help solve similar issues on your interface. Users will appreciate the familiarity and simplicity of those UI patterns.

Creating a sense of Deja Vu helps guide users and instinctively understand how to navigate your site, form, or app. Place labels next to icons rather than having the label appear as the user hovers over the icon. This will only slow them down. Do not assume users are more likely to intuitively move through your site when a commonly used and universally understood symbol or icon is present.

## Effective communication and feedback

Familiar UI patterns will certainly help up your UI design game, but even the most common UI patterns are not foolproof. Think of your interface as in conversation with your user. Communicating with your user throughout every action by providing frequent feedback that validates their actions and nudges them along will go a long way.

Keep users informed with visual cues or simple messaging that clearly indicate whether their actions will lead to the anticipated goal. Make sure your user interface design anticipates user errors with undo actions, text boxes that save information if a form is submitted incompletely or incorrectly. Use messaging as a teachable situation to gently help prevent the error from occurring again.

They should always know where the action was successful or not, their progress, and if further action is needed. This will help create a positive experience where users are confident, satisfied, and reach the end goal.

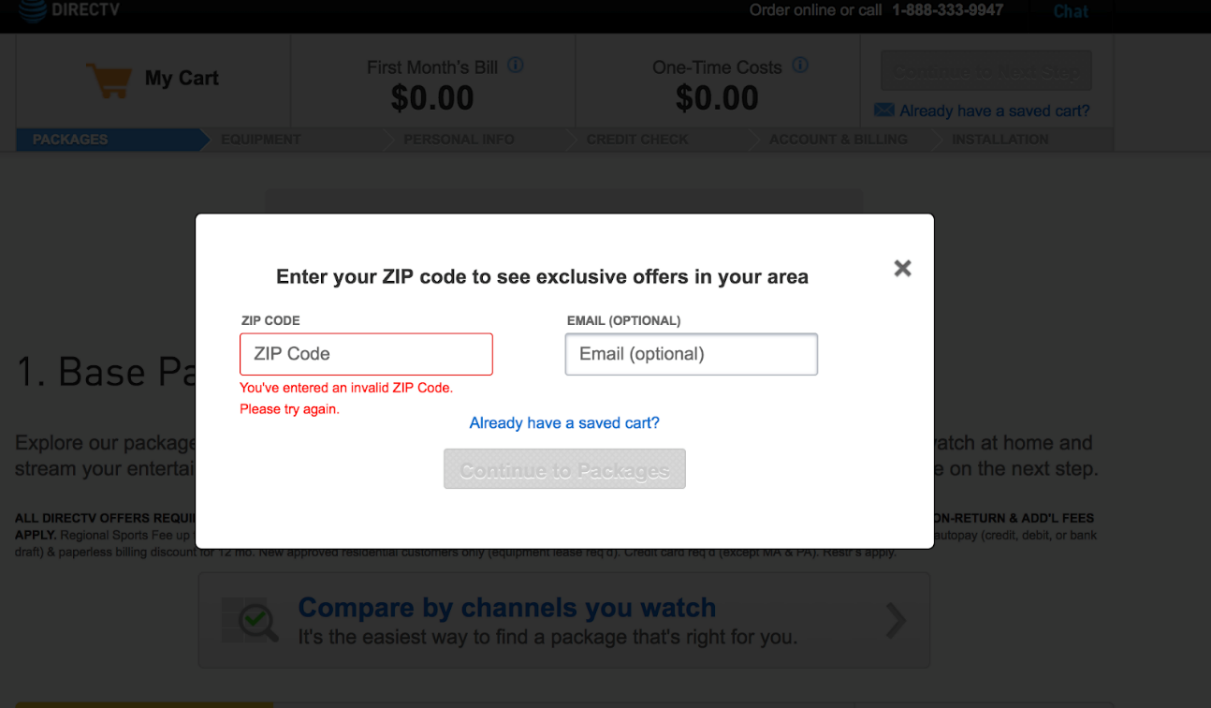


Figure 7: Keep Users Informed With Visual Cues Or Simple Messaging

Stay consistent

Consistency is everything! Once a user understands and learns how to do something, they can achieve it more efficiently and seamlessly next time. Everything from language, layout, colour scheme, navigation buttons, menus, and any other design element should remain the same throughout. These small details will keep the user on task and eliminate unnecessary distractions.

Users will be more comfortable and confident if they are asked to perform the same kind of task multiple times. Make sure similar tasks look familiar and respond predictably.

## Visual hierarchy

A strong visual hierarchy is the crux of an aesthetically successful interface. Similarly, to consistency, every screen, menu, and page should provide the same visual look and hierarchy. One of the most important design elements is reducing the appearance of complexity, regardless of how complex an action itself may be. Maintaining this throughout reinforces a sense of familiarity as they move through subsequent screens. Only add or change design elements if the action is absolutely necessary.

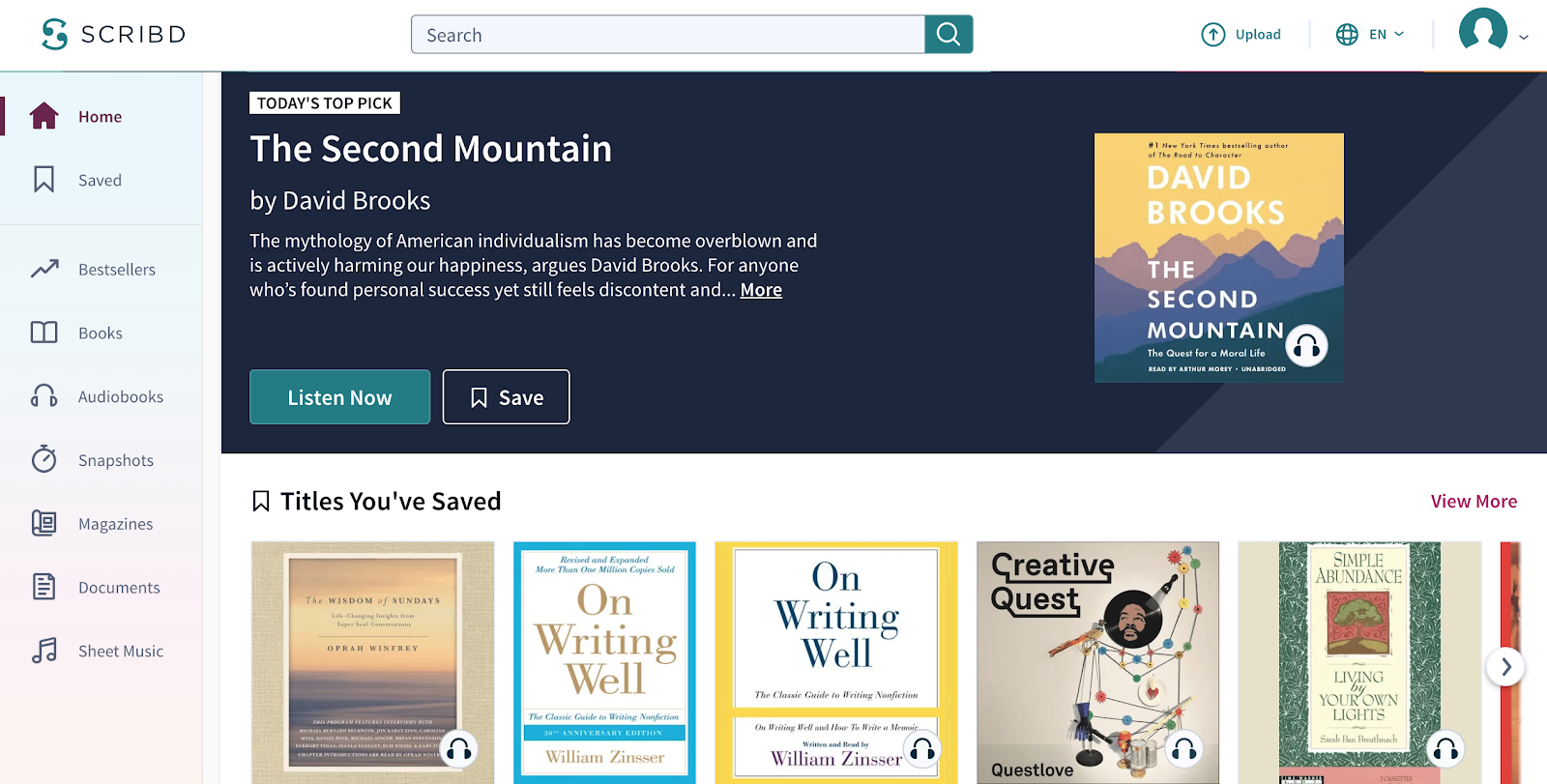


Figure 8: Every Screen, Menu, And Page Should Provide The Same Visual Hierarchy

Empower your user

Consistency and carefully engineered visual hierarchy will empower your user. Once you have built up their confidence and they have become familiar with your interface, you can begin rewarding the user by reducing the amount of guidance and instructions you provide. Oversimplifying can become tedious and overbearing. Incorporate more innovative and abstract ways to move users through tasks. Implementing elements, such as keyboard shortcuts, to accomplish tasks will allow your UI design to remain invisible, and thus, more user-friendly.

Keep it simple and purposeful

UI design works best when it is an invisible guide and considered easy to use and simple by your target audience. When contemplating features or elements, always consider if it will enhance or diminish your users’ experience. Start with what is absolutely essential and ensure that every feature has a purpose. Only include and add features that you can clearly defend the need for.

**QUESTIONS:**

1. Explain the mobile application.

**Answer:**

* 1. **A type of software application that is intended to run specifically on mobile devices.**

1. Explain type of mobile application.

**Answer:**

* 1. **Native apps**
  2. **Hybrid apps**
  3. **Web apps**

1. The different between native apps and hybrid apps

**Answer:**

* 1. **Native apps are built specifically for a mobile device’s operating system.**
  2. **Hybrid apps are web apps that look and feel like native apps.**
  3. **Native apps generally have better performance compared to hybrid apps.**
  4. **Hybrid apps are more economical compared to native apps.**

1. List major mobile operating system in the market:

**Answer:**

* 1. **Android**
  2. **iOS**

1. Programming languages used to develop Android application:

**Answer:**

* 1. **Java**
  2. **Kotlin**

1. Programming languages used to develop iOS application:

**Answer:**

* 1. **Swift**
  2. **Objective-C**

1. IDE to develop Android applications

**Answer:**

**Android Studio.**

1. IDE to develop iOS applications

**Answer:**

**Xcode**

1. Describe Ionic Framework

**Answer:**

**Ionic uses front-end technologies like HTML, CSS, JavaScript, and Angular for application development. Using web technologies, Ionic helps to build cross-platform mobile applications with a single codebase. Basically, it allows web developers to create web pages that are run inside a device’s browser instance called WebView. WebView may come as a plugin, and it is essentially an application component that renders web pages and displays them as a native application.**

1. Describe React Native

**Answer:**

**React Native is a JavaScript framework for developing mobile applications that can run natively on both Android and iOS. It is based on ReactJS, developed at Facebook, which is a declarative, component-based framework for developing web user interfaces. Like ReactJS, React Native uses JSX, an XML markdown language for developing UIs which replaces HTML and CSS. From JSX, Ul components are compiled into native platform-specific components which create a fast and familiar experience for end-users.**

**REFERENCE:**

1. [https://www.techopedia.com/definition/2953/mobile-application-mobile-app\](https://www.techopedia.com/definition/2953/mobile-application-mobile-app%5C)
2. <https://clevertap.com/blog/types-of-mobile-apps/>
3. <https://www.lifewire.com/what-is-ios-1994355>
4. <https://recombu.com/mobile/article/what-is-ios-and-what-does-ios-stand-for>
5. <https://www.androidauthority.com/android-app-development-1128595/>
6. <https://semaphoreci.com/blog/what-is-react-native>
7. <https://www.educative.io/blog/flutter-tutorial>
8. <https://dev.to/sm0ke/firebase-short-introduction-for-beginners-24d8>
9. <https://xd.adobe.com/ideas/principles/>
10. <https://www.getcloudapp.com/blog/ui-design>